# Quantum Algorithms for Event Generation

Snowmass Computational Frontier Workshop
CompF6: Quantum Computing

CWB, W. de Jong, B. Nachman, D. Provasoli

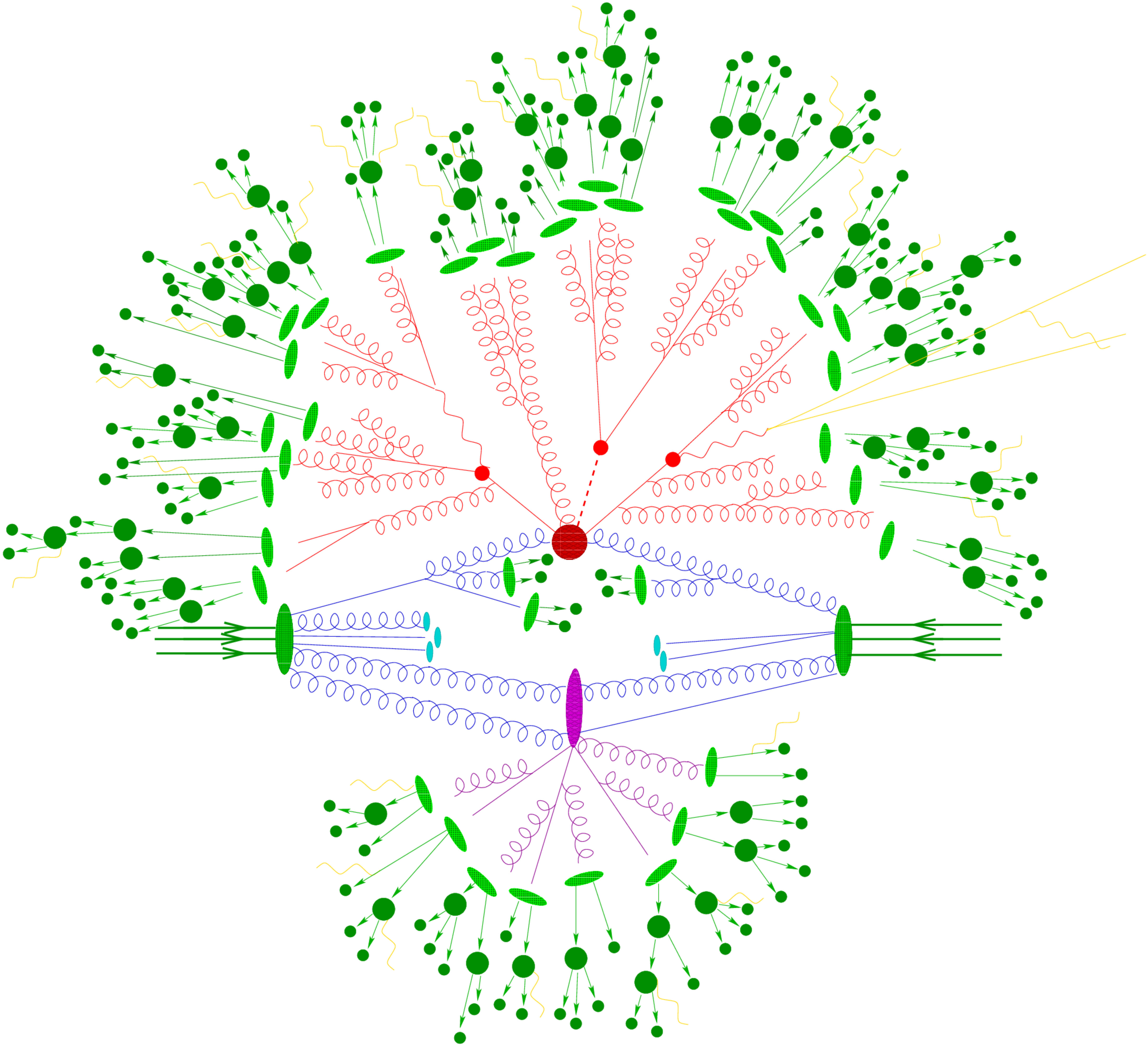
1904.03196 [quant-ph]

# There are different ways in which quantum computing can make theoretical predictions for scattering cross sections

- Simulate the full time evolution of the relevant Quantum Field Theory
  - In principle possible in polynomial time
  - Requires very large resources, that are probably not available for a very long time
- Simulate the only a subset of the full QFT
  - Simulate the low energy behavior (EFT simulation)
  - Simulate only a part of the traditional pieces of a full calculation (Short distance perturbative, Parton Shower, Hadronization, Other effects)
- Will focus on the Parton Shower (Event Generator) in this talk, but idea is very general
- Also LOI by Matchev, Mrenna, Shyamsundar, Smolinsky on similar topic

Quantum Computing for HEP Theory and Phenomenology

K. T. Matchev*, S. Mrenna†, P. Shyamsundar* and J. Smolinsky*



Quantum algorithms for Event Generation

# For collinear emissions from energetic particles squares of amplitudes factor, giving probabilistic interpretation

Markovian process $$\left|A_{n+1}\right|^2 \approx \left|A_n\right|^2 \times P(t)$$

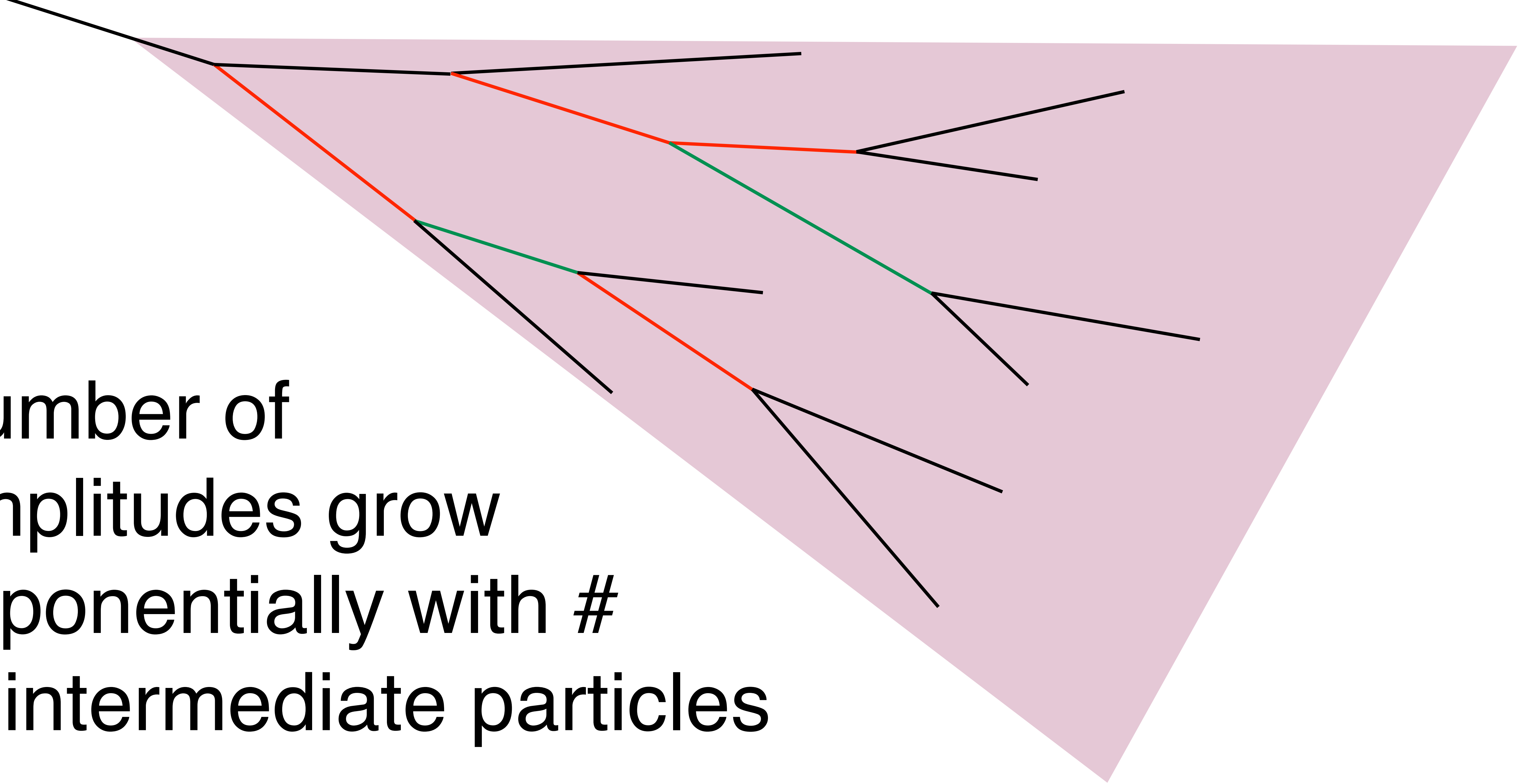
## Two possibilities at each t:

1. Nothing happens (no-branch prob $\Delta$)
2. Emission happens (branch prob $P \times \Delta$)

```
state = initial_state()
for t in 1… N:
    if emission_happens(state):
        n = choose_emitter(state)
        state = new_state(state, n)
write_out(state)
```

…but parton shower is completely based on probabilities, so all quantum mechanical information is lost...

…to get it back, need to compute shower for each possible amplitude…



Number of amplitudes grow exponentially with # of intermediate particles

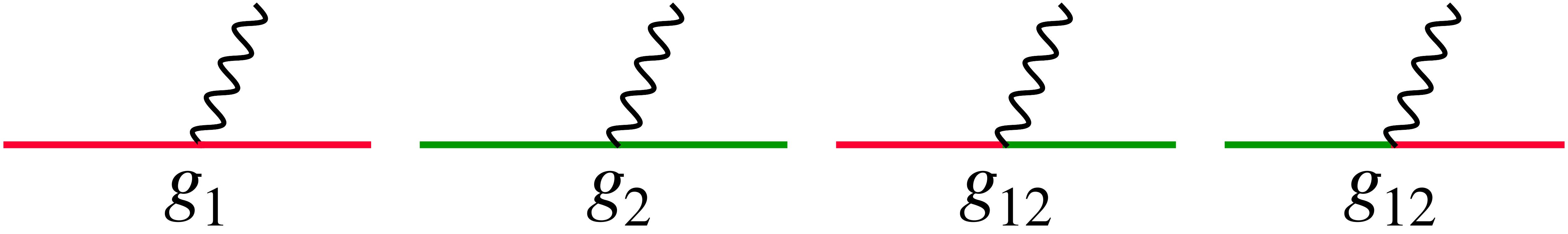
Doing this problem on a classical computer is in general exponentially hard

# A very simple toy model

Yukawa theory with two types of fermions and mixing between them

$$\mathcal{L} = \bar{f}_1(i\partial\!\!\!/ + m_1)f_1 + \bar{f}_2(i\partial\!\!\!/ + m_2)f_2 + (\partial_\mu\phi)^2$$
$$+ g_1\bar{f}_1 f_1\phi + g_2\bar{f}_2 f_2\phi + g_{12}\left[\bar{f}_1 f_2 + \bar{f}_2 f_1\right]\phi$$

Very simple Feynman rules

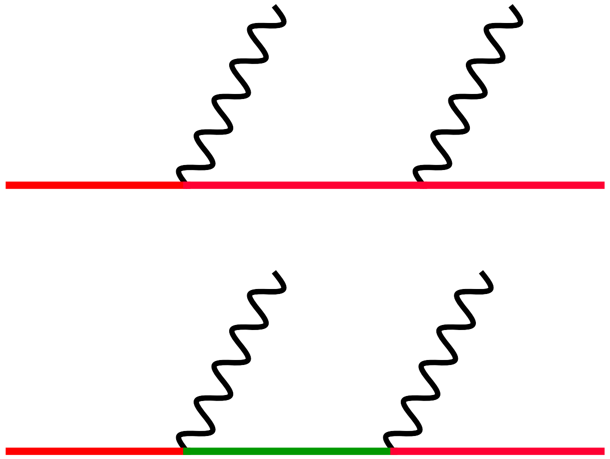


# A very simple toy model

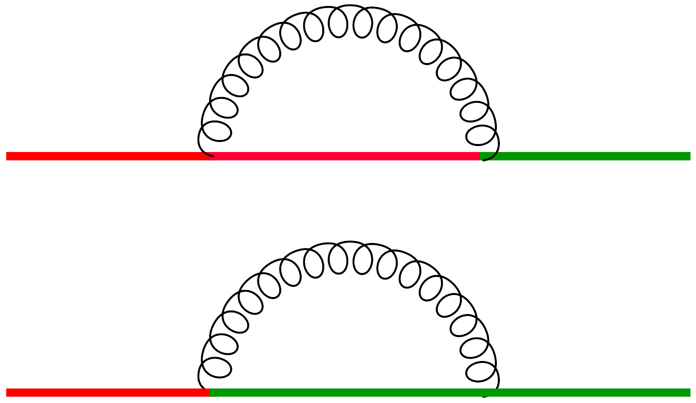
$$
\begin{aligned}
\mathcal{L} =& \bar{f}_1(i\not{\partial} + m_1)f_1 + \bar{f}_2(i\not{\partial} + m_2)f_2 + (\partial_\mu\phi)^2 \\
&+ g_1\bar{f}_1 f_1\phi + g_2\bar{f}_2 f_2\phi + g_{12}\left[\bar{f}_1 f_2 + \bar{f}_2 f_1\right]\phi
\end{aligned}
$$

The mixing $g_{12}$ gives several interesting effects

Different real emission amplitudes give rise to interference



Virtual diagrams give rise to flavor change without radiation



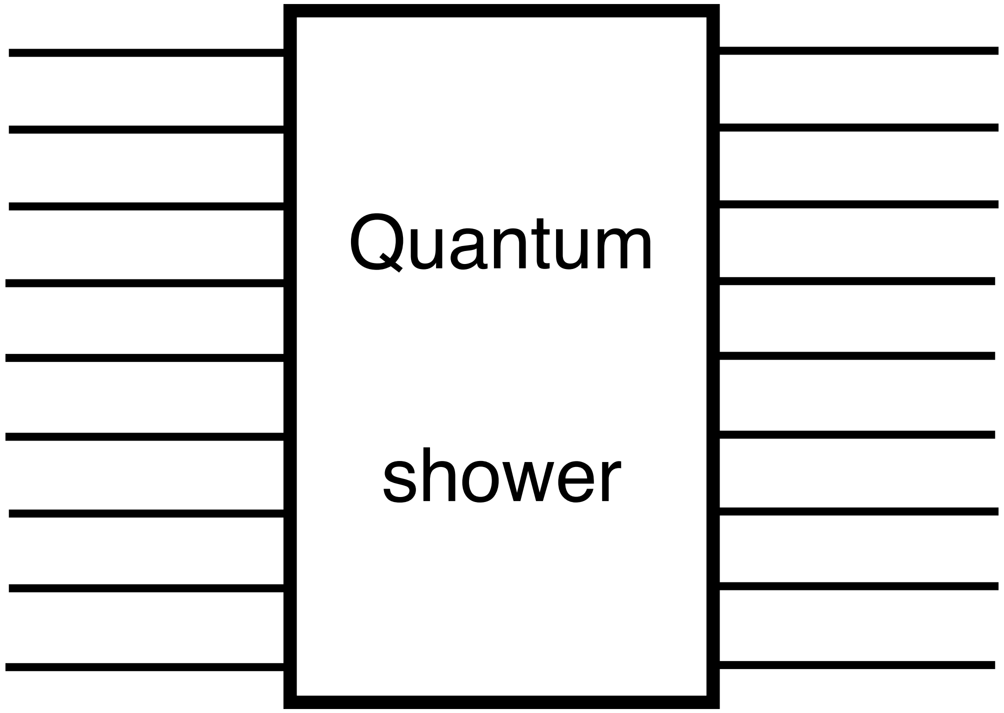
Need to correct both real and virtual effects

Similar to including subleading color

Simulating this model in full generality on classical computer exponentially hard

# A quantum computer can compute the $2^{nf}$ amplitudes using polynomial number of operators

Goal of algorithm is to create superposition of final states with correct relative amplitudes



Quantum

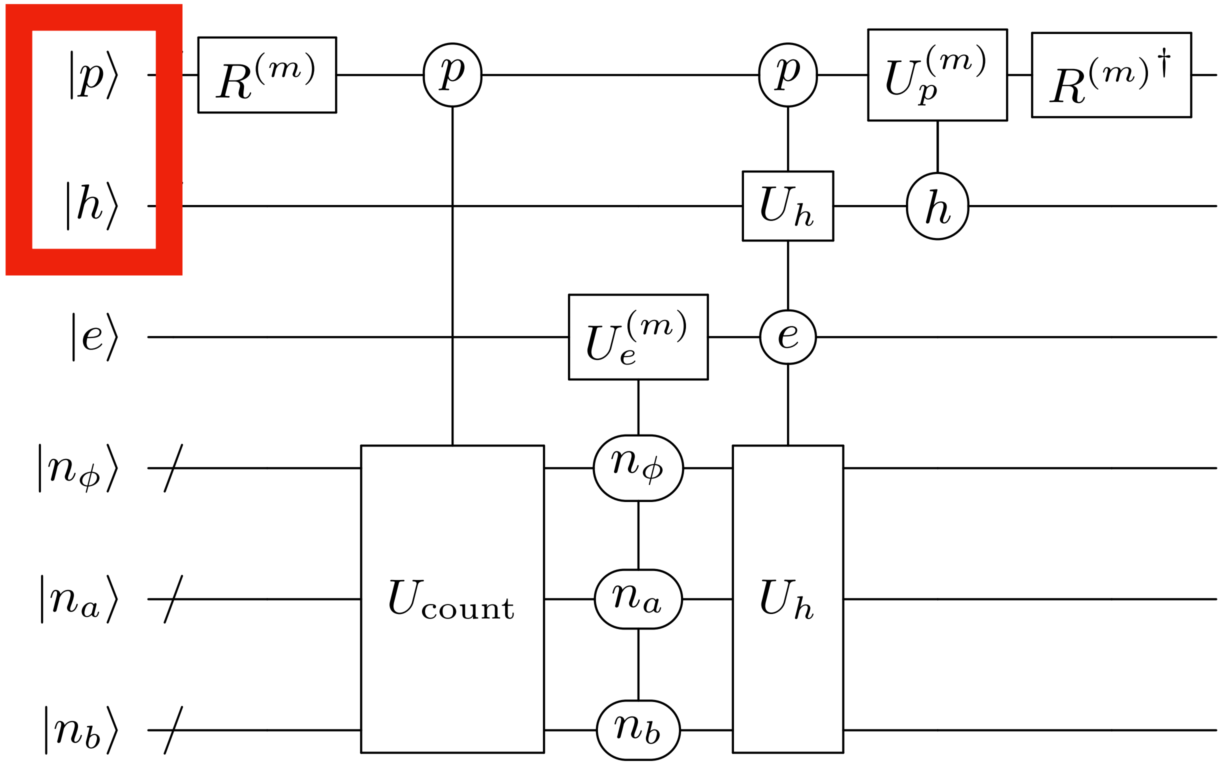
shower

$$|000\ldots0\rangle \rightarrow A_1\,|\Psi_1\rangle + \ldots A_n\,|\Psi_n\rangle$$

Repeated measurements of the final state selects states with probability $|A_i|^2$ ⇒ can be used as true event generator

# A quantum computer can compute the $2^{nf}$ amplitudes using polynomial number of operators

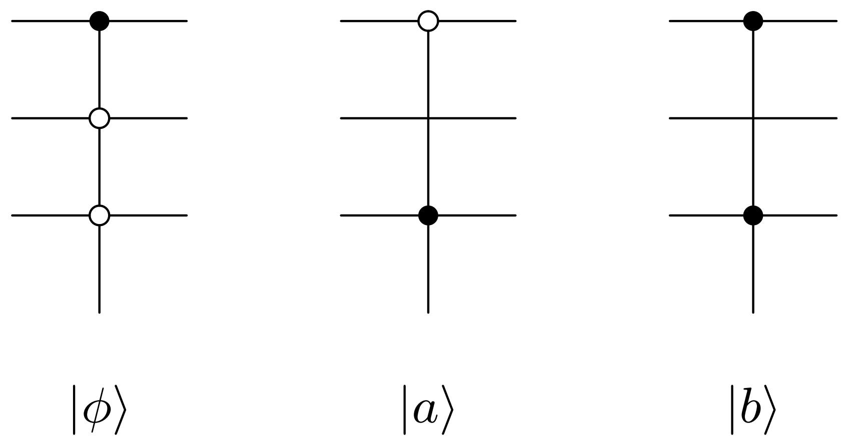
At each discreet time interval, algorithm rotates from $f_1$, $f_2$ basis to $f_a$, $f_b$ basis, performs shower in 4 separate steps, and rotates back to $f_1$, $f_2$ basis



$|n_i\rangle\,,|h\rangle$ : Integer registers

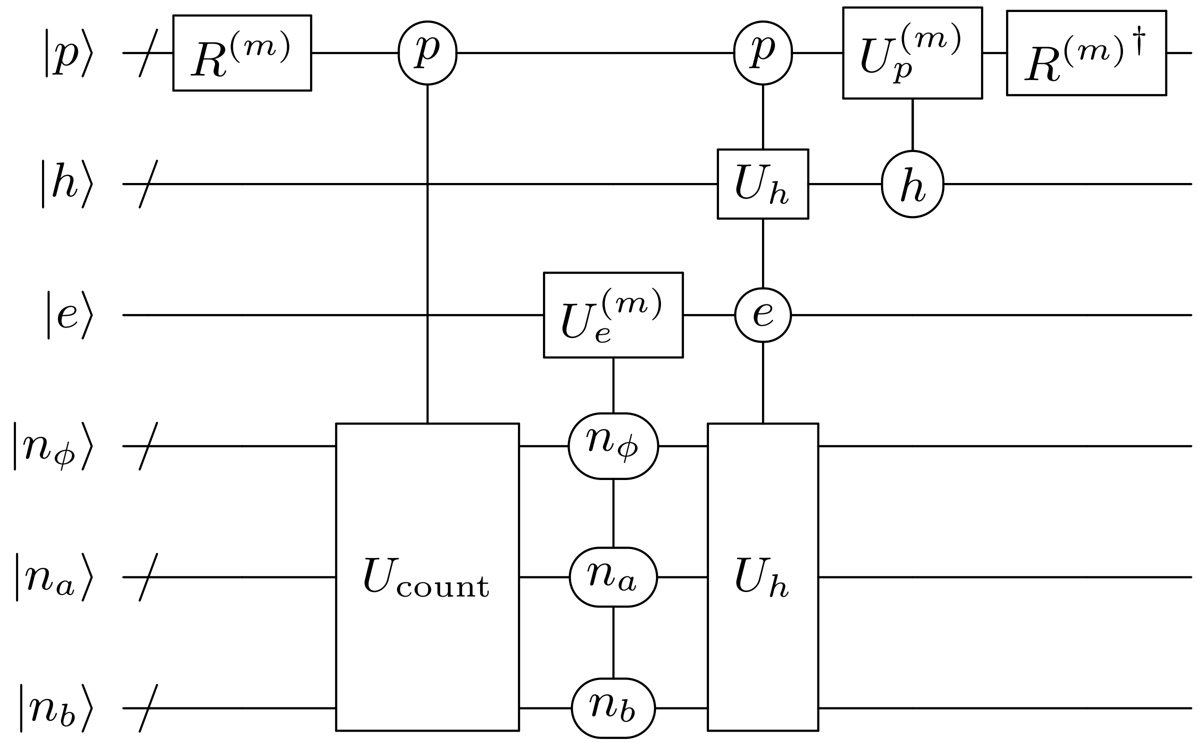
$$|p\rangle_i = \begin{pmatrix}000\\001\\010\\011\\100\\101\\110\\111\end{pmatrix} = \begin{pmatrix}0\\ \phi\\ -\\ -\\ f_1/f_a\\ f_2/f_b\\ \bar{f}_1/\bar{f}_a\\ \bar{f}_2/\bar{f}_b\end{pmatrix}$$

$|e\rangle$ : Boolean value



# A quantum computer can compute the $2^{nf}$ amplitudes using polynomial number of operators

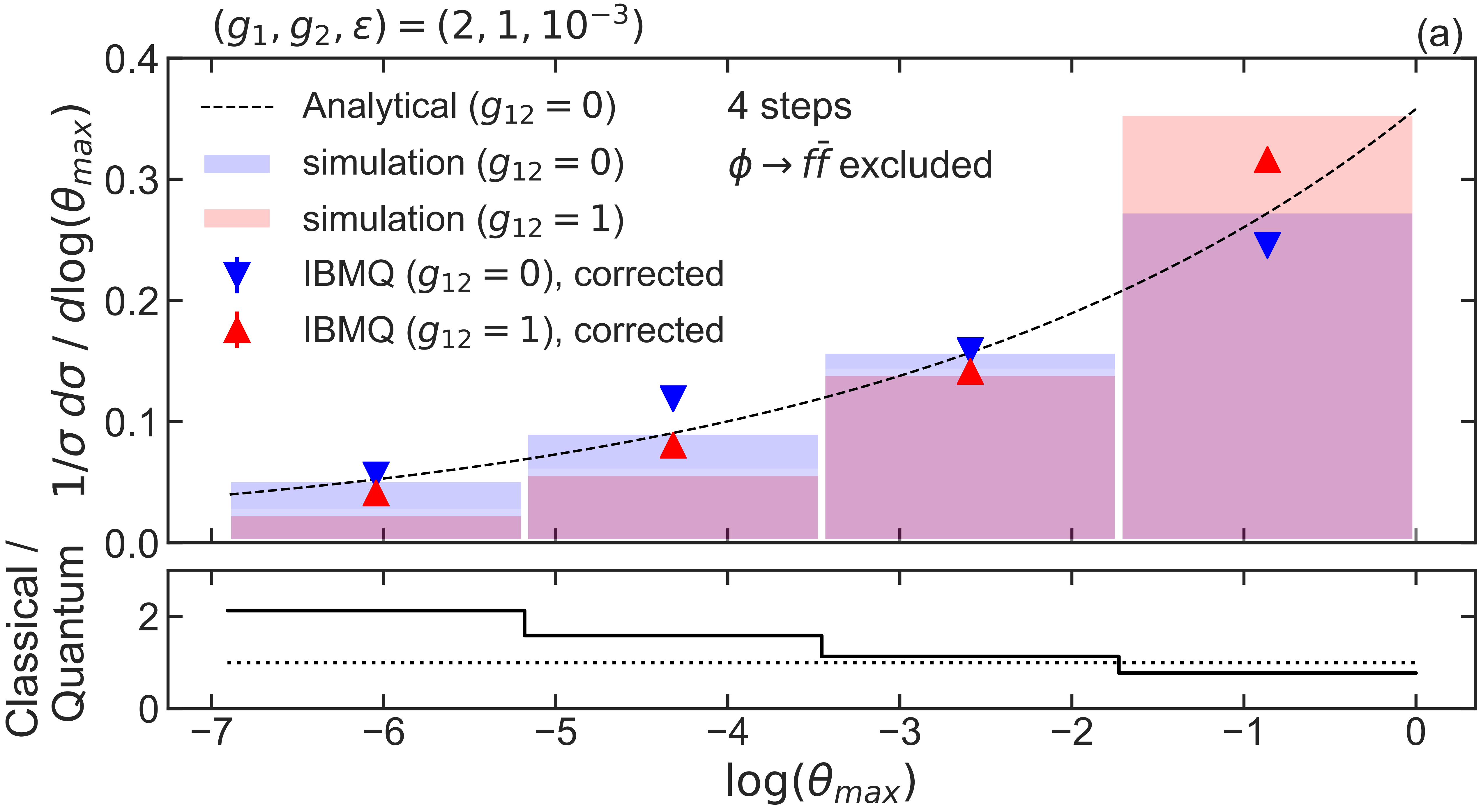
At each discreet time interval, algorithm rotates from $f_1$, $f_2$ basis to $f_a$, $f_b$ basis, performs shower in 4 separate steps, and rotates back to $f_1$, $f_2$ basis



| Operation | Scaling |
|---|---|
| count particles<br>$U_{count}$ | N ln($n_f$) |
| decide emission<br>$U_e$ | N $n_f$ ln($n_f$) |
| create history<br>$U_h$ | N $n_f^2$ ln($n_f$) |
| adjust particles<br>$U_p$ | N $n_f$ ln($n_f$) |

classical algorithms scales as

$$N\,2^{n_f/2}$$


$(g_1, g_2, \varepsilon) = (2, 1, 10^{-3})$
(a)
Analytical ($g_{12} = 0$)
simulation ($g_{12} = 0$)
simulation ($g_{12} = 1$)
IBMQ ($g_{12} = 0$), corrected
IBMQ ($g_{12} = 1$), corrected
4 steps
$\phi \to f\bar{f}$ excluded
$1/\sigma\, d\sigma\ /\ d\log(\theta_{max})$
0.0
0.1
0.2
0.3
0.4
Classical / Quantum
0
2
−7
−6
−5
−4
−3
−2
−1
0
$\log(\theta_{max})$

# There are many things that needs to happen before this becomes truly useful

1. Apply to quantum interference effects of standard model
2. Reduce the circuit depth and required qubits
3. Find ways to make code more robust against noise
4. ……………………………

**But our proof of principle that quantum interference effects in parton showers can be included using quantum algorithms is important first step**

# There are some important questions we need to try to answer in the Snowmass process

- What are the most promising questions QC might provide a breakthrough ultimately

- What might be possible in various different scenarios
[O(100) noisy qubits, O(100) clean qubits, O(1000) clean qubits etc

- Are there any special hardware requirements HEP has [size of system, connectivity etc]

- What kind of collaborations are envisioned between algorithm and hardware developers?